Constraints and Markov chains for a Cognitive Model of Jazz Improvisation: Adding Chords

Kristy Yun and Mariana Montiel, PhD

Undergraduate Research Initiations in Mathematics,
Mathematics Education, and Statistics at Georgia State University
30 Pryor St SW Atlanta, GA 30303

Abstract

Jazz improvisations are composed of repeated rhythmic and melodic patterns. By studying how these patterns come to be can help us to better understand how real-time decisions are made in the context of a given structure. One theory proposes that improvisers rely on memorized patterns of pitch and rhythm and these patterns are inserted into an ongoing improvisation. The competing theory states that improvisers are able to generate notes using algorithms and rules of tonal jazz without the aid of memorized patterns. Previous study analyzed a large corpus of 48 improvised solos by the jazz great Charlie Parker. Results from this study shows that the incidence of patterns present in the Charlie Parker corpus coincides with the current pattern-based algorithm, while Impro-Visor, a rule-based software, does not generate this presence of patterns. In addition to these results and to the current algorithm, the next step is to incorporate chords without letting them dictate the melody and contour of the output of the current algorithm. A possible solution is to use non-homogeneous Markov models and to treat the chords as constraints. There are many applications to these algorithms, that go beyond jazz and even music, given they are based on patterns. Creativity in areas such as gaming can be studied using these algorithms, where players must respond creatively while acquiring certain patterns of response as a result of rules and constraints.

Introduction

When performing with small jazz groups, professional jazz musicians do not read music. Jazz players will choose phrases that seem to the audience to be preordained but are actually being created at that moment. These professional musicians are creating a very intricate form of theme and variation; each aware of their tune and role[1], which is why jazz improvisation serves as an excellent paradigm for studying real-time creativity. Jazz improvisation is also a prototype of the mental activity common to speech recognition and other areas of interest in Artificial Intelligence.

Currently, there are two competing cognitive theories in the study of jazz improvisation: (1) the pattern-based approach and (2) the rule-based approach (see Fig. 1). There

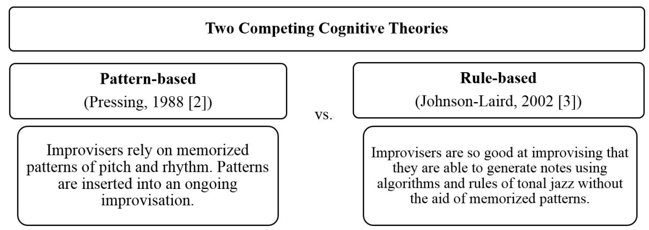

Figure 1. Competing Cognitive Theories

are several improvisation softwares that are based on one of these two approaches. One example of a rule-based software is Impro-Visor, a music notation software designed to help jazz musicians compose and hear solos similar to ones that might be improvised. A previous study explored the two competing theories by analyzing a large corpus of improvisations by jazz great, Charlie Parker. The results from this study (shown in Fig. 2) showed that the percentage of notes that start a 4-interval pattern as a function of the number of times the pattern occurs in the improvisations made by Impro-Visor is not coherent with the Charlie Parker corpus. However, our melody algorithm seems to mirror the Charlie Parker corpus. In order to incorporate the chords while changing the resulting improvisations of our algorithm as little as possible we will be using non-homogeneous Markov chains.

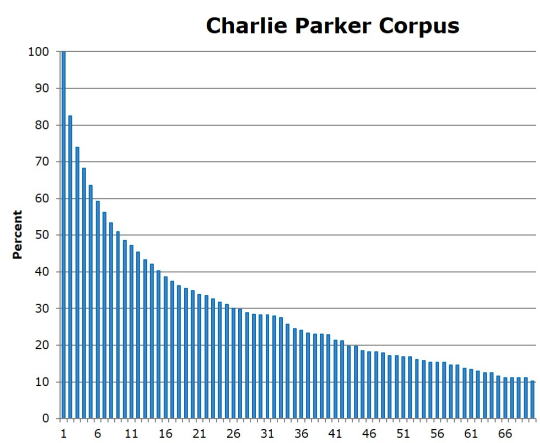

Figure 2 (part 1). Comparison of percentage of notes that start a 4-interval pattern as a function of the number of times the pattern occurs. [2]

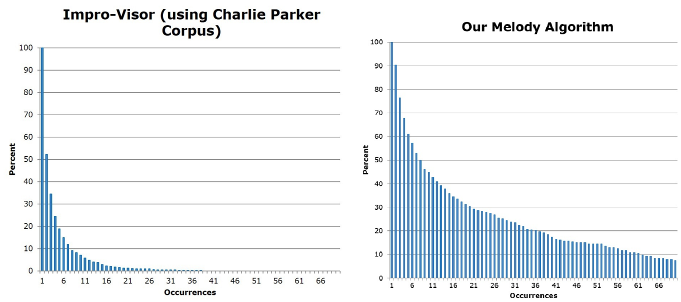

Figure 2 (part 2). Comparison of percentage of notes that start a 4-interval pattern as a function of the number of times the pattern occurs. [2]

Markov Models Markov processes are a popular modeling tool used in content generation, such as text generation, music composition and interaction. The basic principle of the Markov assumption is that future states depend only on the immediate past and not on the sequence of events that occurred before it. Mathematically,
for a sequence $\{q_1, q_2, ..., q_n\}$:

$$p(q_i|q_1, ..., q_{i-1}) = p(q_i|q_{i-1}). \tag{1}$$

Example of Markov Processes[4] Weather prediction is all about trying to guess what the weather will be like tomorrow based on a history of observations of weather. Given Table 1 of arbitrarily picked numbers shown and the automaton generated from this table in Fig. 3, let us try to predict the weather.

For example, given that today is sunny, what is the probability that tomorrow is sunny and the day after is rainy? This translates to:

$$\begin{aligned} & P(q_2 = \text{Sunny}, q_3 = \text{Rainy} \,|q_1 = \text{Sunny}) \\ = & P(q_3 = \text{Rainy} \,|q_2 = \text{Sunny}, q_1 = \text{Sunny}) \times \\ & P(q_2 = \text{Sunny} \,|q_1 = \text{Sunny}) \\ = & P(q_3 = \text{Rainy} \,|q_2 = \text{Sunny}) \times \\ & P(q_2 = \text{Sunny} \,|q_1 = \text{Sunny}) \\ = & (0.05)(0.8) \\ = & 0.04 \end{aligned}$$

This probability can also be obtained by moving through the automaton in Fig. 3 and multiplying the probabilities as you go.

Results

In Pachet's paper *Finite-length Markov processes with constraints*, he shows that control constraints can be compiled into a new Markov model that is statistically equivalent to the initial one. "This yields the advantage of retaining the simplicity of random walk, while ensuring that control constraints are satisfied."[5] These results can be applied to our current melody algorithm to maintain the probabilities of the originally generated "improvised" output while incorporating the chords as constraints.

		Tomorrow's Weather		
		Sunny	Rainy	Foggy
Today's Weather	Sunny	0.8	0.05	0.15
	Rainy	0.2	0.6	0.2
	Foggy	0.2	0.3	0.5

Table 1. Arbitrarily picked probabilities for the weather.

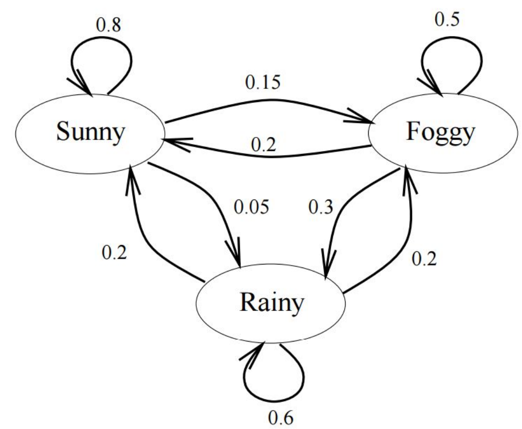

Figure 3. Automaton generated from Table 1.

Our goal is to generate a non-homogeneous Markov model, represented by a series of transition matrices. In order to show how constraints can be compiled into a non-homogeneous Markov model, we will look at a melody generation example with a simple constraint. Let the constraint be that all the 4-note melodies generated must end in C.
Consider a Markov model M estimated from the sequences in Fig. 4.
The prior vector M:

$$\begin{array}{c} \begin{array}{ccc} C & D & E \end{array} \\ \left(\begin{array}{ccc} 1/3 & 1/2 & 1/6 \end{array} \right) \end{array}$$

where the entries are found using the input melodies in Fig. 4. For instance, to find the probability of C we first take the total number of notes in each melody, which in this case is 6 notes for each. Then, of those six notes two are C's and thus we get the probability $2/6 = 1/3$.

The transition probabilities of M can also be generated from the two input melodies.

$$\begin{array}{c|c} & \begin{array}{ccc} C & D & E \end{array} \\ \begin{array}{c} C \\ D \\ E \end{array} & \left(\begin{array}{ccc} 0 & 2/3 & 1/3 \\ 3/5 & 1/5 & 1/5 \\ 1/2 & 1/2 & 0 \end{array} \right) \end{array}$$

For example when C goes to D, we can see from our melodies that the total possible transitions starting with C are:

- C goes to D (first melody)
- C goes to E (second melody)
- C goes to D (second melody).

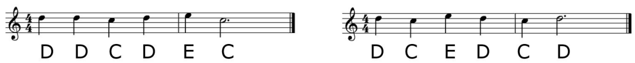

Figure 4. Two simple input melodies used to estimate M.

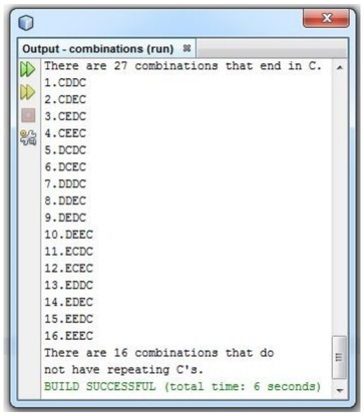

Figure 5. All possible combinations of 4-note melodies satisfying the constraint.

Of the three possible transitions, 2 out of 3 end in D. Hence the probability of going from C to D is $2/3$.

Using a simple program to generate all possible combinations of 4-note melodies (see Fig. 5), we get 12 possible non-zero probabilities shown in Table 2.

The probabilities of the 4-note melodies ending in C can be found using our prior vector M and the transition probabilities. For instance,

$$\begin{aligned} p_M(CDDC) &= p_M(C)\, p_M(D|C)\, p_M(D|D)\, p_M(C|D) \\ &= (1/3)(2/3)(1/5)(3/5) = 2/75. \end{aligned}$$

After the generation of these primary matrices, the first step in our process is to make the induced constraint satisfaction problem arc-consistent. Arc-consistency consists in propagating the constraints in the whole CSP, through a fixed-point algorithm that considers constraints individually [6]. For our example, arc-consistency removes C and E from the domain of V_3 yielding the following domains:

$$\begin{array}{ccccccc} V_1 & \xrightarrow{K_1} & V_2 & \xrightarrow{K_2} & V_3 & \xrightarrow{K_3} & V_4 \\ \{\text{C,D,E}\} & & \{\text{C,D,E}\} & & \{\text{D}\} & & \{\text{C}\} \end{array}$$

where K_i is the state transition between $Z^{(i-1)}$ to Z^i and V_i is the state. This ensures that during any random walk, there will not be a situation in where an item with no continuation is chosen. Now the next step is to extract the matrices from the domains.

Using Pachet's algorithm [5]:

- Initialization:
  $Z^(0) \leftarrow M_0$ (the prior probabilities of M),
  $Z^(i) \leftarrow M, \forall i = 1, ..., L-1$ (the transitions).
- For each $a_k \in A$ removed from the domain of V_i:
  ${Z_{j,k}}^{(i)} \leftarrow 0, \forall j = 1, ..., n$ (set the k-th column to zero).
- All forbidden transitions in the binary constraints should also be removed from the matrices:
  ${Z_{j,k}}^{(i)} \leftarrow 0, \forall i, j, k$ such that $B_i(a_j, a_k) =$ *false*.

Melodies	Probabilities	Melodies	Probabilities
CDDC	2/75	DEDC	3/100
CDEC	1/45	DDEC	1/100
CEDC	1/30	ECDC	1/30
DCDC	3/25	ECEC	1/72
DCEC	1/20	EDDC	1/100
DDDC	3/250	EDEC	1/120

s = 416/1125

Table 2. The 12 4-note melodies satisfying the control constraint and their probabilities in M, where the sum of probabilities for these sequences is s.

we maintain the following matrices:

$$Z^{(0)} = (1/3 \quad 1/2 \quad 1/6) \qquad Z^{(1)} = \begin{pmatrix} 0 & 2/3 & 1/3 \\ 3/5 & 1/5 & 1/5 \\ 1/2 & 1/2 & 0 \end{pmatrix}$$

$$Z^{(2)} = \begin{pmatrix} 0 & 2/3 & 1/3 \\ 0 & 1/5 & 1/5 \\ 0 & 1/2 & 0 \end{pmatrix} \qquad Z^{(3)} = \begin{pmatrix} 0 & 0 & 0 \\ 3/5 & 0 & 0 \\ 1/2 & 0 & 0 \end{pmatrix}$$

Lastly, we build the final transition matrices $\tilde{M}^{(i)}$ to $\tilde{M}$ by a simple right-to-left process in order to back propagate the perturbations in the matrices induced by individual normalization, starting from the right-most one.[5]

To do this, we first normalize individually the last matrix $Z^{(L1)}$. We then propagate the normalization from right to left, up to the prior vector $Z^{(0)}$. The elements of the matrices $\tilde{M}^{(i)}$ and the prior vector $\tilde{M}^{(0)}$ are given by the following recurrence relations:

$$\tilde{m}_{j,k}^{(L-1)} = \frac{z_{j,k}^{(L-1)}}{\alpha_j^{(L-1)}}, \quad \alpha_j^{(L-1)} = \sum_{k=1}^{n} z_{j,k}^{(L-1)}$$

$$\tilde{m}_{j,k}^{(i)} = \frac{\alpha_k^{(i+1)} z_{j,k}^{(i)}}{\alpha_j^{(i)}}, \quad \alpha_j^{(i)} = \sum_{k=1}^{n} \alpha_k^{(i+1)} z_{j,k}^{(i)} \quad 0 < i < L-1$$

$$\tilde{m}_k^{(0)} = \frac{\alpha_k^{(1)} z_k^{(0)}}{\alpha^{(0)}}, \quad \alpha^{(0)} = \sum_{k=1}^{n} \alpha_k^{(1)} z_k^{(0)}$$

Using the above relations, we get the following transition matrices for our example.

- $i = L-1 = 3$

$${\alpha_1}^{(3)} = \sum_{k=1}^{3} {z_{1,k}}^{(3)} = 0 + 0 + 0 = 0$$

$${\alpha_2}^{(3)} = \sum_{k=1}^{3} {z_{2,k}}^{(3)} = 3/5 + 0 + 0 = 3/5$$

$${\alpha_3}^{(3)} = \sum_{k=1}^{3} {z_{3,k}}^{(3)} = 1/2 + 0 + 0 = 1/2$$

$$\tilde{M}^{(3)} = (\tilde{m}_{j,k}^{(3)}) = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

- $i = 2$

$${\alpha_1}^{(2)} = \sum_{k=1}^{3} {\alpha_k}^{(3)} {z_{1,k}}^{(3)} = (0, 3/5, 1/2) \bullet (0, 2/3, 1/3) = 17/30$$

$$\alpha_2{}^{(2)} = \sum_{k=1}^{3} \alpha_k{}^{(3)} z_{2,k}{}^{(3)} = (0, {}^3\!/_5, {}^1\!/_2) \bullet (0, {}^1\!/_5, {}^1\!/_5) = {}^{11}\!/_{50}$$

$$\alpha_3{}^{(2)} = \sum_{k=1}^{3} \alpha_k{}^{(3)} z_{3,k}{}^{(3)} = (0, {}^3\!/_5, {}^1\!/_2) \bullet (0, {}^1\!/_2, 0) = {}^3\!/_{10}$$

$$\tilde{M}^{(2)} = (\tilde{m}^{(2)}_{j,k}) = \begin{pmatrix} 0 & \frac{3/5 \times 2/3}{17/30} & \frac{1/2 \times 1/3}{17/30} \\ 0 & \frac{3/5 \times 1/5}{11/50} & \frac{1/2 \times 1/5}{11/50} \\ 0 & \frac{3/5 \times 1/2}{3/10} & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 12/17 & 5/17 \\ 0 & 6/11 & 5/11 \\ 0 & 1 & 0 \end{pmatrix}$$

By similar computations for $i = 1$ and $i = 0$, we conclude with the following transition matrices:

$$\tilde{M}^{(0)} = ({}^{185}\!/_{832} \quad {}^{999}\!/_{1664} \quad {}^{295}\!/_{1664})$$

$$\tilde{M}^{(1)} = \begin{pmatrix} 0 & 22/37 & 15/37 \\ 85/111 & 11/111 & 5/37 \\ 85/118 & 33/118 & 0 \end{pmatrix}$$

$$\tilde{M}^{(2)} = \begin{pmatrix} 0 & 12/17 & 5/17 \\ 0 & 6/11 & 5/11 \\ 0 & 1 & 0 \end{pmatrix} \qquad \tilde{M}^{(3)} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

Discussion

This is an efficient approach to control Markov generation for control constraints that will:

- guarantee that the generated sequences satisfy these constraints
- follow the probability distribution of the initial Markov model.

We can see that the final transition matrix $\tilde{M}$ maintained the same probability distribution as the prior vector M.

$$\tilde{M}^{(0)} = (\tfrac{185}{832} \quad \tfrac{999}{1664} \quad \tfrac{295}{1664}) = (0.22 \quad 0.60 \quad 0.18)$$

$$M = ({}^1\!/_3 \quad {}^1\!/_2 \quad {}^1\!/_6) = (0.33 \quad 0.50 \quad 0.17)$$

Table 2 shows that the $\tilde{M}$ probabilities of all possible solution sequences, where these probabilities are equal to the initial probabilities, to the constant multiple factor$\alpha^{(0)}$.

These algorithms do not have to be genre specific to jazz, given that they are based on the actual patterns of a corpus. There have been implementations with classical music, fiddling, blue grass, etc. It is thought that this algorithm might even transcend music and be used to study creativity in areas such as gaming, where improvisation is seen to play a significant role given that players have to respond creatively while acquiring certain patterns of response as a result of rules and constraints.

Melodies	Probabilities	Melodies	Probabilities
CDDC	61/969	DEDC	25/353
CDEC	16/305	DDEC	11/466
CEDC	24/305	ECDC	24/305
DCDC	83/293	ECEC	2/61
DCEC	36/305	EDDC	11/466
DDDC	10/353	EDEC	6/305

Table 3. The probability of the set of solution sequences in $\tilde{M}$. The ratio of probabilities is constant.

Presently, we are working on incorporating chords into the algorithm by using this method. My contribution in this research project consisted in finding this technique and showing its relevance for this important next step in developing the improvisation software.

Acknowledgments

First and foremost, I have to thank my research supervisor, Dr. Mariana Montiel. Without her assistance and dedicated involvement in every step throughout the process, this paper would have never been accomplished. I would like to thank you very much for your support, guidance, patience, and above all, mentorship.

I would also like to show gratitude to my Neurophysics research group, including Dr. Mukesh Dhamala, Dr. Martin Norgaard, and Kiran Dhakal for sharing my interest and excitement during the course of this research. Without the opportunity given to me by Dr. Dhamala to work alongside Kiran to take fMRI data from jazz musicians, I would not have been able to work with Dr. Montiel and Dr. Norgaard more closely on this project.

Also, thank you to Dr. Paco Gomez for helping out with the last parts of the computation.

References

[1] "Jazz Improvisation." *A Passion for Jazz! Music History & Education.* http://www.apassion4jazz.net

[2] Pressing, J. (1988). *Improvisation: Methods and model.* In J. A. Sloboda (Ed.), Generative processes in music (pp. 129-178). Oxford, UK: Oxford University Press.

[3] Johson-Laird, P. N. (2002). *How jazz musicians improvise. Music Perception.*, 19, 415-442.

[4] Resch, Barbara, *Hidden Markov Models: A Tutorial for the Course Computational Intelligence.* http://www.igi.tugraz.at/lehre/CI

[5] Pachet , Pierre Roy , Gabriele Barbieri, *Finite-length Markov processes with constraints*, Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, July 16-22, 2011

[6] C. Bessiere, E. C. Freuder, and J.-C. Regin. *Using inference to reduce arc consistency computation.* In Proc. of the IJCAI95, pages 592598. Morgan Kaufmann, 1995.